Neural Machine Translation

Danish Shaikh
M.Tech (ECE)
shaikhm@iisc.ac.in

Abstract

Neural Machine Translation aims at building a single neural network, unlike the traditional statistical machine translation, that can be jointly tuned to maximize the translation performance. Addition to the neural network, there is also a need for attention models. In this report, I have discussed the results of seq2seq models with various different attention mechanisms. The dataset I have used is from the WMT14 machine translation task.

1 Introduction

The recent few years have witnessed a huge surge in the research of Neural Machine Translation (NMT). The proposed model captures the dependencies in long sentences by not fixing it into a fixed-length vector. In addition, I have introduce an extension to the model which learns to align and translate jointly. Each time it soft-searches for a set of positions in a source sentence where the most relevant information is concentrated. The model then predicts a target word based on the context vectors associated with these source positions and all the previous generated target words. I have used the dataset from the WMT14 machine translation task for EN-DE and EN-HI and vice-versa. Below table is the highest BLEU scores of different target and source languages for attention model.

Source	Target	BLEU Score
English	German	0.15
German	English	0.13
English	Hindi	0.17
Hindi	English	0.16

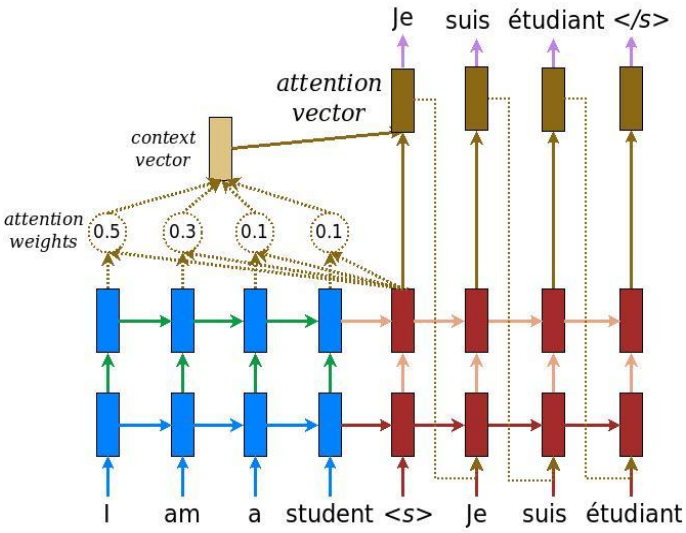
Table 1: BLEU Scores

2 Seq2Seq Model with Attention

I have started with the vanilla seq2seq model. Then applied attention mechanisms to it. The attention computation happens at every decoder time step and feed the output in the next time-step. It consists of the following stages:

- The current target hidden state is compared with all input states to derive attention weights.
- Based on the attention weights, compute a context vector as the weighted average of the input states.
- Combine the context vector with the current target hidden state to yield the final attention vector.
- The attention vector is fed as an input to the next time step (input feeding).

This is illustrated in the below figure.



3 Assignment Tasks

The Assignment consists of following task:

3.1 Task:

The neural network model is trained for various encoder-decoder attention mechanisms:

1. Additive Attention
2. Multiplicative Attention
3. Scaled dot-product attention
4. Key-value Attention Finally, we want to implement Self-Attention Mechanism.

This steps we have to perform for EN-DE Translation and EN-DE Translation.
The hyperparameters that I tweaked to get highest BLUE score were batch size, embedding dimensions, Hidden Size.

4 Implementation

I have used ML libraries such as tensorflow, nltk, keras, etc. for dataset, pre-processing the data and embedding the words. Furthermore, I have used references for building encoder-decoder functions and attention mechanism.

- Task:
    1. Download the WMT14 EN-DE and EN-HI parallel-corpus dataset (as mentioned in the guidelines) and import using nltk in python.
    2. Extract sentences and remove those sentences which has ¿20 words.
    3. Create vocabulary containing unique words from the corpus.
    4. For Embedding Task, vocabulary is first chosen from the source language.
    5. After getting the embeddings, pass them to encoder and decoder.
    6. Train the model using 2 different RNNs. One for Encoder and other one for decoder.
    7. For the different attention mechanisms, pass the arguement accordingly.
    8. Minimize the loss function while applying stochastic gradient descent algorithm.
    9. Validate for hyperparameters on the development task (as mentioned in the guideline).
    10. Obtain BLEU score.

Equations of various hidden layers for calculating gradient of loss function is as follows:

$$\alpha_{ts} = \frac{\exp\left(\text{score}(\boldsymbol{h}_t, \bar{\boldsymbol{h}}_s)\right)}{\sum_{s'=1}^{S} \exp\left(\text{score}(\boldsymbol{h}_t, \bar{\boldsymbol{h}}_{s'})\right)} \quad \text{[Attention weights]} \tag{1}$$

$$\boldsymbol{c}_t = \sum_s \alpha_{ts} \bar{\boldsymbol{h}}_s \quad \text{[Context vector]} \tag{2}$$

$$\boldsymbol{a}_t = f(\boldsymbol{c}_t, \boldsymbol{h}_t) = \tanh(\boldsymbol{W}_{\boldsymbol{c}}[\boldsymbol{c}_t; \boldsymbol{h}_t]) \quad \text{[Attention vector]} \tag{3}$$

Here, the function 'score' is used to compared the target hidden state with each of the source hidden states, and the result is normalized to produced attention weights (a distribution over source positions). There are various choices of the scoring function; popular scoring functions include the multiplicative and additive forms given in Eq. (4). Once computed, the attention vector is used to derive the softmax and loss. This is similar to the target hidden state at the top layer of a vanilla seq2seq model.

$$\text{score}(\boldsymbol{h}_t, \bar{\boldsymbol{h}}_s) = \begin{cases} \boldsymbol{h}_t^\top \boldsymbol{W} \bar{\boldsymbol{h}}_s & \text{[Luong's multiplicative style]} \\ \boldsymbol{v}_a^\top \tanh\left(\boldsymbol{W}_1 \boldsymbol{h}_t + \boldsymbol{W}_2 \bar{\boldsymbol{h}}_s\right) & \text{[Bahdanau's additive style]} \end{cases} \tag{4}$$

Different types of attention mechanisms:

1. Additive Attention.
2. Multiplicative Attention
3. Scaled Dot-Product Attention
4. Key-value Attention
5. Self-Attention

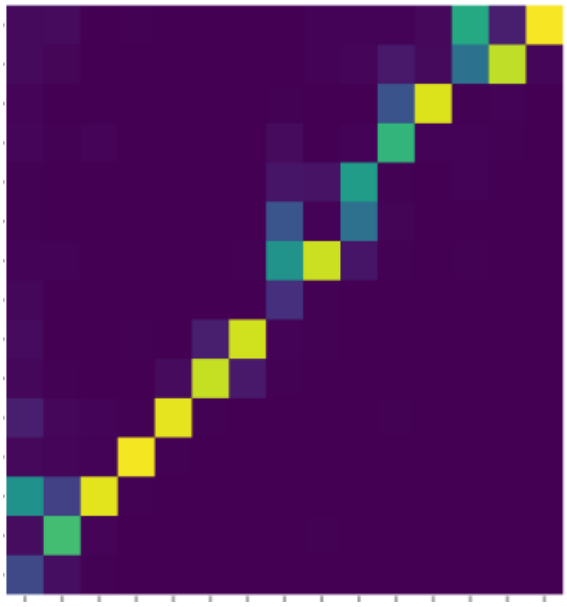
5 Problem of Underfitting and Overfitting

Depending on the dataset we should select hyperparameters to avoid underfitting and overfitting.

1. For this model, I have observed that if we increase embedding size beyond 300, the performance starts to decrease in terms of BLEU Score.
2. To avoid overfitting, I have considered dropout probability=0.2

6 Analysis

The Attention Visualization for Multiplicative is shown below:



I have validated the model keeping batch-size=32, Hidden Size=300 and Embedding Dimensions = 300.
The best model, in terms of BLEU score, is of self-attention mechanism. Different attention mechanisms gives different BLEU scores for a particular dataset.
All the results are tabulated below (Refer Table 2:) :

7 Conclusion

- This seq2seq with attention model using 2 different RNNs for encoder and decoder learns weights from the corpus and translate from source language to target language.

- This model can be further optimized if we use synonyms of words to represent the translated output differently. As BLEU score doesn't take that into account. Thereby increasing the BLEU Score.

8 Github

All the files are uploaded in the Github repositories. The link is provided here.

9 References

1. Bahdanau, Dzmitry, Kyunghyun Cho, and Yoshua Bengio. "Neural machine translation by jointly learning to align and translate." arXiv preprint arXiv:1409.0473 (2014).

2. Neural Machine Translation by Google Reasearch Blogpost. Link is here

3. Vikram Bhatt's (classmate) guidelines

Source	Target	Attention Mechanisms	BLEU Score
English	German	Additive	0.09
English	German	Multiplicative	0.11
English	German	Scaled-Dot Product	0.12
English	German	Self-Attention	0.15
English	Hindi	Additive	0.10
English	Hindi	Multiplicative	0.13
English	Hindi	Scaled-Dot Product	0.13
English	Hindi	Self-Attention	0.17

Table 2: BLEU scores.